xp_enhancedFeatures

V1.0.1 – 2024-09-16 by Septirage

Guide for Admin

Requirements

To use the xp_aspectManager plugin you will need :

- Nwnx4
- xp_bugfix (include in the nwnx4 package)

Installation of xp_enhancedFeatures :

Put **xp_enhancedFeatures.dll** and **xp_enhancedFeatures.ini** in your nwnx4 folder.

Add the **nwnx_enhancedfeatures_spell.nss** file in your module (or import it with xp_enhancedfeatures.erf), this files contain all the functions you can use and will be detailed latter in this documentation.

Look at our website (https://septirage.com/nwn2/d/Plugins?id=xpenhancedfeatures) to have an overview.

List of xp_enhancedFeatures.ini options

xp_enhancedFeatures use at least one ini file. This ini file can also define other ini files, as we will see below

loglevel:

Adjust the log verbosity level as needed (from 0 to 5).

EnhanceStoreRetrieve:

Set to 1 to enhance the possibility of **StoreCampaignObject** and **RetrieveCampaignObject**.

By default, these functions are limited to Creatures and Items. However, with this patch, they will also work for Placeables, Doors, Triggers, Waypoints, Lights, and PlacedEffects.

Note: If you use **SQLStore/RetrieveObject**, these functions will also be able to store the additional object types in a database, as they rely on **Store/RetrieveCampaignObject**.

PatchDestroy:

Set to 1 to fix the **DestroyObject** function.

This patch ensures that the function works correctly on Areas and prevents memory leaks when deleting an object with an inventory (such as a Creature or Chest).

Previously, if an object with inventory was deleted, the object itself would be deleted, but not all items in it.

This would leave the items inaccessible and cause a memory leak.

This patch fixes this issue, and the game will now correctly delete all items in the inventory when the object is destroyed.

Note : It's important to note that this bug occurs not only on explicit nwnscript calls to DestroyObject, but also whenever an object is destroyed. One particularly significant example of this is when a player logs out, the PC is destroyed, causing memoryleak each time. Obviously, this patch will fix all occurrence of the bug.

PatchSetTag:

Set to 1 to fix the **SetTag** function.

Bug Explanation:

Without the patch, due to a bug in the engine, when you change an object tag, this object will be returned by GetObjectByTag(newTag) AND GetObjectByTag(oldTag).

PatchCloneArea:

Set to 1 to patch the **CreateInstancedAreaFromSource** function.

The Patch will allow to use this function with all area, even the one created by a previous call of CreateInstancedAreaFromSource. It will also add the ID of original Area as CreatorID to the new one.

FixItemPropertyLoad:

Set to 1 to enable the fix.

Bug Explanation:

Without the patch, if a player logs in with items that provide bonuses equipped, the bonuses will not be removed as they should be when the item is unequipped. With the patch, this issue is fixed and the bonuses will be properly removed when the item is unequipped.

FixSaveThrow:

Set to 1 to enable the fix.

Bug Explanation:

Without the fix, Will and Reflex saving throws were not correctly subject to automatic failure. This meant that if the total roll was higher than the DC and the roll was a natural 1, the save still succeeded instead of automatically failing as it should have.

FixWildShapeUsage :

Set to 1 to enable the fix.

Bug Explanation:

Without the fix, if the PC has the "Extra Wild Shape" feat, "*Oaken Resilience"* and "*Elephant's Hide"* will restore charges of "Wild Shape" instead of consuming them.

FixDodgeVSDamageTypeRemoval :

Set to 1 to enable the fix.

Bug Explanation:

Without the fix, if the PC unequip an item with a property "Dodge AC Bonus vs Damage Type", the game will remove this specific bonus but also the same amount of Dodge AC. This will repeat each time, causing the DodgeAC value of the creature to reduce again and again.

FixDecreaseSpellOnBonusLoss:

Explanation:

By default, for Sorcerer/Bard type class, the game will reduce the number of SpellUsage each time you remove a bonus (buff, or item) that grant Spells or SpellcastAbility.
As it is normal if none of the spell was used, it can be seen as a problem in other case.

Example :

You have a Sorcerer with 5 lvl 1 spell per day with a ring that grant you +1 lvl1 spell. If you unequip it, your number of available spell will reduce to 4. It's normal. But if you equip it and unequip it again, your number of available spell will reduce to 3 etc.

This option allow you to choose how to react for this.

Remove this option or set it to 0 to keep the base game behavior.
Set it to 1 to reduce only if the new maxSpellPerDay is lower than your current available spell.
Set it to 2 to never reduce this number.

DisableTumbleAC:

Set it to 1 to Disable the bonus AC given by Tumble Skill score.

DisableSpellCraftSave:

Set it to 1 to Disable the bonus to SpellSave given by SpellCraft skill score.

MonkWeaponList:

Here you can list the BaseItem will see as monk weapons.
For info, base game list is : “36 40 50 59 78”

AreaTypeBitX:

Those are specials Option. You will be able to link a bit of Area Type (that can be accessed with xp_aspectManager), with a name.
Base game options are :

AreaTypeBit0 = INTERIOR
AreaTypeBit1 = SUBTERRAN
AreaTypeBit2 = NATURAL

Those names will be used in the rule system, detailed bellow.

SpeedFeatsFile:

Set it to the name of the ini file where you will list the Rules for SpeedFeat.
Rule system will be detailed bellow. Plugin came with an example file : SpeedFeatEFF.ini

SkillFeatsFile:

Set it to the name of the ini file where you will list the Rules for SkillFeat and Ability-Skill link.
Rule system will be detailed bellow. Plugin came with an example file : SkillFeatEFF.ini

WeaponFinesseFile:

Set it to the name of the ini file where you will list the Rules for WeaponFinesse.
Rule system will be detailed bellow. Plugin came with an example file : WeaponFinesseEFF.ini

OnGoldChangeScript:

Set a script name that will be called on a new event added by this plugin : OnCreatureGoldChange. It will be called each time the gold owned by a creature will change. It will also allow you to modify this change if you want.
More detail about this specific script bellow.

GlobalOnClientEnterScript:

Set a script name that will be called on the OnClientEnter event for every area. It will happens **after** the area-specific OnClientEnter.

This can be used if you want to execute a specific script each time a PC load any area of you module.

StartTransitionScript:

Set a script name that will be called on the new event added by this plugin : OnCreatureStartTransition.

It will be called when a creature will try a transition (by using a door or a transition trigger). It will also allow you to cancel this transition if you want.

More detail about this specific script bellow.

Rules System Files

The rule system will allow you to easily add new feat or capability. Changing hardcoded part of the game. Currently, three part of the game work with the Rule system. More will be added with time.

Lot is common between all of those files, but some stuff are specific.

Common:

Each rule start with a **[SpecificNameRuleX]**. X being a number. The plugin will start by parsing the rule 1 and continue as long as it can find the next. So please, be sure to have contiguous number in your rules.

Some field will understand math operation, other will understand Boolean operation. Allowing you to really customize it to your need.

Allowed symbol for Boolean operation : | (or) , & (and) , ^ (XOR) , ! (not) and Parenthesis ().

Allowed symbol for Math operation : + (sum) , - (substraction), * (multiplication), / (division) , // (integer division) , % (modulus) and parenthesis ().

Math operation also allow two specific functions : ClassLevelSum and ClassLevelMax

ClassLevelSum : Will sum the pc level of each of listed class. You can put 1 or as many class as you want as parameters. Usage example : ClassLevelSum(5, 45) will sum the monks and sacred fist level of the creature.

ClassLevelMax : Same format as level sum but will only return the greatest level of listed class.

Each rules can contain the following fields :

Feat : Boolean field. Here you can choose which feat will activate the rules. You can put the feat number. The field will understand Boolean operation so, if you want to activate the rule if the PC have feat 1 or feat 2 and no feat 3, you can write something like that : *Feat = (1 | 2) & !3*.

Area : Boolean field. Here you can choose in which kind of Area the rule will apply. You will need to use the AreaType name listed in the main ini file. For example, if you want that the rule will apply only on a exterior and natural area you can write something like that : *Area = !INTERIOR & !SUBTERRAN & NATURAL*

Bonus : Mathematical field. More detail on specific part.

Extra : Boolean field. Little more specific field. Here you will be able to make your rule depend of the activation (or non-activation) of previous rules. It will also provide you the possibility to use Special Check. For now, only special check available is "MONKPOWER". To check if a previous rule is activated, just write RuleX , with x= the number of your specific rule.
/!\ this will always be RuleX, regardless of the SpecificName of the rule.
For example, if your rule must activate only if the rule 1 is ok, the 2 not ok and your pc have MONKPOWER available you can write something like that : Extra : Rule1 & !Rule2 & MONKPOWER.

Special :

MONKPOWER : Will be true if you play a monk that don't "loose" his power by wearing an armor for example.

SpeedFeatEFF:

Note : Using this file will remove all base game feats. The example file reimplement them in the RuleSystem so you can easily just add news one.

The speedFeat ini file can start with a **[General]** part with the field **CalculationType**.
Set the CalculationType value to Sum or Factor to choose how the different speed bonus will be calculated.

The base game use "factor mode". That mean that, if you have two feats that give you a bonus of 10%, you will end up with a speed "factor" of 1.21. (1 * 1.1 * 1.1 = 1.21), like if you have a single bonus of 21%. If you use some, you will have a result of 1.20 (1+ 0.1 + 0.1).

Example :

```
[General]
CalculationType = Factor
```

SpecificName : Your rules name must be in the [SpeedRuleX] format.

Bonus : Will be used to determine the % of bonus you want to apply. In a factor for (0.2 for 20%). A negative value will reduce the speed. A positive one will increase it.

Specific Rules : There is two specific kind of rule in this file :

[MinSpeedRuleX] : Allow you to set the Minimal speed factor possible. Order is important as only the first valid one will be used.

The Bonus field is renamed Min here.

[MaxSpeedRuleX] : Allow you to set the Maximal speed factor possible. Order is important as only the first valid one will be used.

The Bonus field is renamed Max here.

For those two specifics rules type, if you choose to use Extra with RuleX test on it, those will refer to SpeedRuleX.

In this file, for each rules, the only mandatory ones are the Bonus one. (or Min/Max for MinSpeedRuleX/MaxSpeedRulesX types).

SkillFeatEFF:

The skillFeat ini file can start with a **[General]** part with the field **RemoveBaseSkillFeatRules**.
Set the RemoveBaseSkillFeatRules value to 1 in order to remove all base game skill feat bonus. Else the base game feat bonus will apply normally.
Remove the whole part or set the value to 0 to keep the game feats.

Example :
[General]
RemoveBaseSkillFeatRules = 1

SpecificName : Your rules name must be in the [SkillRuleX] format.

Bonus : Will be used to determine the value of bonus you want to apply. A positive value will increment the skill, a negative one will reduce it. Note that you can only use whole number and no number with decimal point for this one.

Skill : A specific field. Only one number allowed : the skill that will be concerned by the rule.

For each **[SkillRuleX]** mandatory fields are **Bonus** and **Skill**. Other are optional.

This file also allow you to set a second set of rules. The SkillAbility one. Those rules will allow you to dynamically change the ability used for a skill. For example.. Allow to use Strength for intimidate when you have the right feat.

SpecificName: Your rules name must be in the [SkillAbilityX] format.

Skill : A specific field. Only one number allowed : the skill that will be concerned by the rule.

Bonus : Not allowed for those rules.

Extra : Here, this field don't allow the usage of RuleX

Ability : The ability that must be used. Use only one of those possible values : STR, DEX, CON, INT, WIS, CHA.

For each **[SkillAbilityX]** rules, **Skill** and **Ability** fields are mandatory. **Feat** and **Area** fields are optionals

The order of the **[SkillAbilityX]** rules is important as only the first valid one will be used.

WeaponFinesseEFF:

The weaponFinesse ini file can start with a **[General]** part with the field **RemoveBaseWpnFinesseRules**.
Set the RemoveBaseWpnFinesseRules value to 1 in order to remove all base game weapon finesse rules. Else the base game weapon finesse rules will apply normally.
Remove the whole part or set the value to 0 to keep the game behavior.

Example :
[General]
RemoveBaseWpnFinesseRules = 1

SpecificName : Your rules name must be in the [WpnFinesseRuleX] format.

ItemType : A specific field. Set a list of number, separated by space. List the ItemType allowed to work with Weapon Finesse by this rule.

CheckSize : Set to 1 if this rule need to check the size of the weapon, compared to the size of creature. If set to 0 or absent, the rule will not check the size.

Bonus : Not allowed for those rules.

Extra : Here, this field don't allow the usage of RuleX

For each **[WpnFinesseRuleX]** rules, **ItemType** is mandatory. **Feat**, **Area**, **Extra** and **CheckSize** fields are optional.

If at least one rule is ok, the usage of WeaponFinesse will be allowed for the item. If you don't remove the Base Weapon Finesse Rules, they will be tested if none of your rules allow this item.

Specific Scripts

See the examples scripts distributed with the plugin for details.